# 5.4 Implementing Artificial Intelligence Behaviors in a Virtual World

Implementing Artificial Intelligence Behaviors in a Virtual World

Brian Krisler
Raytheon BBN Technologies
bkrisler@bbn.com

Michael Thome
Raytheon BBN Technologies
mthome@bbn.com

**Abstract**. In this paper, we will present a look at the current state of the art in human-computer interface technologies, including intelligent interactive agents, natural speech interaction and gestural based interfaces. We describe our use of these technologies to implement a cost effective, immersive experience on a public region in Second Life. We provision our Artificial Agents as a German Shepherd Dog avatar with an external rules engine controlling the behavior and movement. To interact with the avatar, we implemented a natural language and gesture system allowing the human avatars to use speech and physical gestures rather than interacting via a keyboard and mouse. The result is a system that allows multiple humans to interact naturally with AI avatars by playing games such as fetch with a flying disk and even practicing obedience exercises using voice and gesture, a natural seeming day in the park.

## 1.0 INTRODUCTION

When artificial intelligence (AI) entities exist in a virtual world, their existence is intended to fill out the scene and help to make the interaction more believable. When you walk down a street, you are typically not alone. There are other people around, in the shops and passing by in their vehicles. These individuals are all going about their personal activities, independent to you, yet they also have a role in your actions and goals. They can cut you off, slow you down or distract you from your intended goal. Should the need arise; you can also interact with these physical world agents using typical forms of communication such as speech and gesture. However in a virtual world, the extent of interaction allowed between humans and AI is often limited to a small vocabulary with an even smaller set of AI driven characters. These limitations produce an interaction that often feels shallow or fake - it is missing the level of realness provided by the *extras* encountered in the physical world.

**Figure 1: To demonstrate our architecture, we implemented a virtual dog in Second Life.**

Improving the realness of AI within a virtual world requires that the entities have the ability to act as entities within the physical world, where they seek and perform goals. To provide goal-seeking behavior to an AI and have them exhibit the expected behaviors associated with the actions performed towards the attainment of those goals, we used an off-the-shelf rules engine to generate the behaviors of AI within a virtual environment. The rules engine provides the ability to assign actions and goals to an entity, where each action is based upon a set of facts. These facts are derived from the world and include the current state of the AI and any ongoing communications with other avatars. The occurrence of these internal and external events helps to drive the creation and attainment of goals, providing a level of realism not typically seen in virtual worlds such as Second Life.

As a demonstration of our technology, we implemented a rules-driven dog avatar (**Error! Reference source not found.**) within Second Life. When in the virtual world, a user avatar can freely interact with our dog. The AI dog can perform the typical actions exhibited by dogs at any local park such as sit and play dead. To extend the level of realism, our dog also performs various idle behaviors. When left alone, the dog will wander about the scene, lie down or bark at random objects. When summoned, he will stop his idle behavior and come to the person calling him.

## 2.0 CURRENT STATE OF AI IN VIRTUAL WORLDS

A cursory review of AI within the history of gaming reveals two common threads:

- Tightly constrained interactions with the AI, as seen in NetHack [1] , ELIZA [2], PARRY [3, 4] and Jabberwacky [5].
- Or rigid operational situations with no moral ambiguity and expectations that constrain the AI behaviors.

Each of these threads can produce a convincing AI within their well-defined context, however in a situation such as a human/AI conversation, their limitations become quickly apparent.

Within a virtual world, the AI (bots), while based upon the common threads, present relatively simple behaviors that are programmed with basic scripted responses or rely upon external engines to provide the intelligence. They are usually embodied as fancy virtual objects that are easily distinguished from human-operated avatars.

In Second Life, for instance, there are a vast number of objects, scripted with simple rule engines to provide primitive behaviors. These range from shop keepers, greeters, and bartenders, to autonomous animals and pets, to art installations that react to surrounding activity, but they all share the feature in that their appearance diverges significantly from human avatars – they don't move, talk, or animate like people.

Another form of AI within a virtual world (VW) is an automated system that uses real avatars, essentially acting as expert systems controlling the avatar as a human operating a VW viewer would via a modified but otherwise standard viewer program or through a library that emulates such a program. However, these are most often used not to simulate intelligent beings in the world, but instead are used as surrogates for humans, allowing them to perform actions not allowed for scripted objects. For example, one of the first uses of these systems in Second Life was to automatically search the virtual world for cheap land for sale and snap it up. Another form of bot within a VW is a "model bot." Model bots have no AI per se, but rather use their embodiment to model clothing, and other looks realistically.

Finally, there have been a few, mostly dead-end, experiments in using embodied AI systems to interact with people on a peer basis. Generally, these have not been convincing and, as a result, have been relegated to research niches, and have never gained any widespread adoption.

## 3.0 OUR APPROACH TO AI IN A VIRTUAL WORLD

To represent AI within a virtual world, we modeled our approach on the way a human interacts with and communicates within the world around them. When a human confronts a world, either real or virtual, they are driven by a series of goals and actions. As they progress about the world and encounter others, the immediate goals can change, with new goals and actions constantly getting created, executed, and completed. For example, the simple act of walking down a street to meet a group of friends (Figure 3) could incur multiple sub-goals along the way. Each of these sub-goals consists of its own actions that are independent of the primary goal. The insertion and execution of multiple sub-goals is an important part of modeling how a human operates with the world. Each new sub-goal operates without impacting the main goal. However in existing virtual environments, AI interact with the world independent of the actions around them. Because they contain a limited set of behaviors, they are restricted in the number of actions they can perform, preventing them from displaying new and unexpected

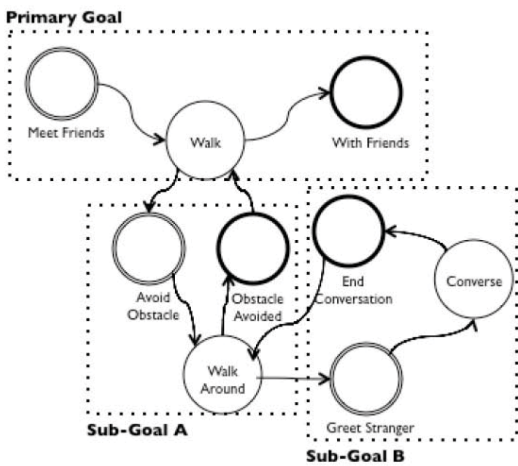


**Figure 3. Goals consist of actions and sub-goals, where the insertion and execution of sub-goals occurs independent of the primary goal.**

behaviors in unique situations, making their actions look scripted and artificial.
To improve the look and feel of AI in a virtual world, we approached the problem through improved input modalities and improved behavior modeling.

## 3.1 Improved Input Modalities

As part of our design, we wanted to create a more seamless interaction between the physical and virtual world. A critical component of this interaction is the ability to communicate with an artificial agent as seamlessly as communicating in the physical world. When a user interacts with a virtual world, they rely upon the keyboard and mouse to act as an interface between themselves and the world. However this interface provides an unnatural interaction, where all thoughts and actions must get conveyed using written text and graphical interface widgets. This form of interaction does not easily lend to the interface disappearing [6], and misses the mark on user experience enrichment.
To attain a more seamless interaction, we extended the standard modalities used to interact with a virtual world. In our system, we added two new input methods: natural speech and physical gestures.

Through the incorporation of speech via a wireless headset and physical gestures using a time-of-flight camera [7], users are provided the means to interact with entities in the virtual world as they would interact in the physical world. Now instead of having to type *come here* into a console, a user can speak the phrase using natural language or wave their arm in a summoning motion to request the attention of a virtual avatar.
If a user wants to play catch with their virtual dog, simply executing a throwing motion could result in an object getting thrown in the virtual world.

By providing the ability to speak real commands and interact using physical actions the *gulf of execution* [8] between the user and the virtual world is reduced, leading to a more enriching user experience.

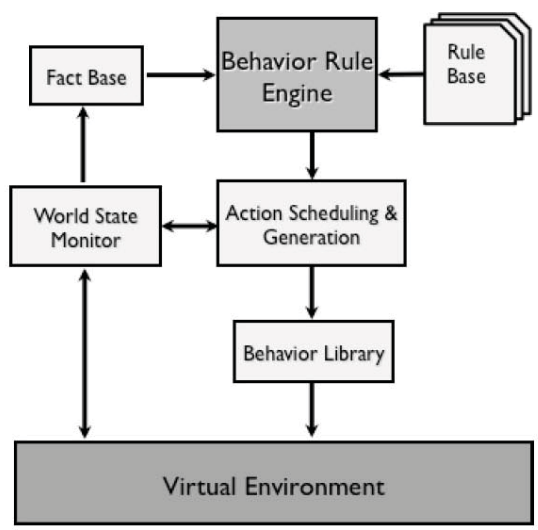


**Figure 2. To create believable AI that exhibit goals and actions, we developed a behavior rules engine based upon facts and rules.**

## 3.2 Improved Behavior Models

To produce a more realistic AI model where an agent can react to unexpected events, we based our design upon a behavior rules engine. A rules engine generates actions and behaviors in response to a combination of facts and rules (Figure 2) where facts characterize the current state of the agent and environment and the rules define the desired actions and behaviors that should occur for an observed factual state.
Each agent starts with an initial set of facts defining their basic characteristics such as name, age and gender. These initial facts set the stage for future actions and behaviors. When the agent enters a recognized world state, the facts of the world are used in conjunction with the initial facts about the agent to determine the appropriate action for the current situation.

Each of these fact-behavior definition pairs are defined as a rule within a rules base. Each rule consists of two parts: the conditional and the action. When the conditionals are met, the action is triggered. For example, an agent could have a rule for responding to a question, such as "What is your name?" where the conditional part of the rule consists of: *is the speaker looking at me and is the speaker within conversation distance*.

To determine the appropriate action, the rule engine utilizes the *Rete* algorithm [9]. The Rete algorithm is a pattern-matching algorithm commonly used in production systems to determine the best rule to fire for a given factual state. For example, an AI agent could approach the speaker's avatar in the virtual world in response to a request to *come here*.

## 4.0 IMPLEMENTATION

To implement our design, we combined a suite of open source tools with proprietary code and custom behavioral rules (Figure 4). These components were integrated the public, and unmodified Second Life virtual world in the middle.

## 4.1 The Artificial Intelligence

Interaction with the virtual world required a method for controlling avatars within the simulator. Typically, avatars within Second Life are controlled via a human operator. This operator communicates with the world through a client interface using inputs, such as a keyboard and mouse to manipulate the avatars behavior. However for our AI backed avatar, we had to replace the human-centric input mechanisms with generated behaviors.

To simulate a goals and actions based behavior, based upon the current state of the virtual world, we leveraged the open source rules engine *Drools* [10].

Drools is a Rete algorithm backed rules engine developed by the JBoss Community. Using Drools, we were able to create a set of rules to attain the behaviors we desired from our avatar.

In our implementation of a virtual dog, we required two sets of behaviors, *Active Behaviors* and *Idle Behaviors*. An Active Behavior is a behavior that occurs when the avatar is reacting to an input from the world, such as a request from another avatar. An Idle Behavior on the other hand is a

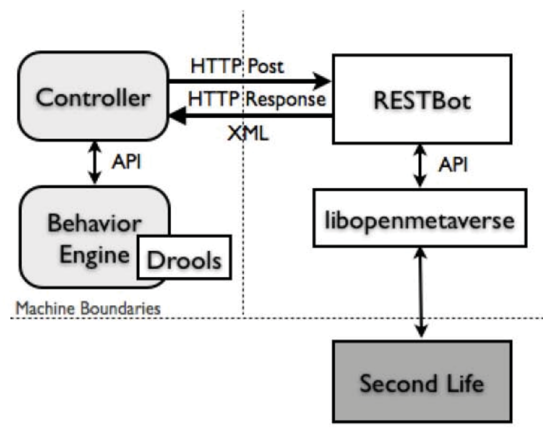


**Figure 4. To implement the architecture, we relied on a suite of open source tools. All of the white boxes represent the open sources tools, and the light gray boxes are custom software. While the protocols support a distributed architecture, for our setup we ran all processes on a single computer.**

behavior that occurs during idle time, when there is no direct interaction with any other avatars. In our implementation, the idle behaviors include actions such as bark, lie down, and take a drink of water.

When the AI avatar is present in the virtual world (i.e. alive), the rules engine executes an idle behavior loop. This loop will generate random behaviors from a list of known idle behaviors, producing a dog that will wander about the scene, performing various actions. If another avatar interacts with the dog, the idle loop stack is cleared, allowing the active behavior to take precedence. Employing this approach causes the dog to stop whatever idle action it was performing and react to the other avatar. By defining a library of 10-15 actions, we were able to create an AI dog in Second Life that exhibited behaviors typical of a dog in the physical world.

To apply our actions and behaviors to an avatar in Second Life, we used the open source toolkit RESTBot [11, 12]. RESTBot is a REST based framework built on top of the open source toolkit, libopenmetaverse [13]. Running as a lightweight HTTP server, it listens for POST commands containing the desired interactions with the virtual world. When a command is received, it translates the HTTP message into virtual world actions. Upon processing of the command, any expected results are passed back to RESTBot, which in turn repackages the results in XML and passes the XML content back to the controller, all within a single HTTP transaction.

The remaining piece of the architecture was linking the Drools engine to RESTBot. To make this link, we implemented a simple controller based upon the command pattern that maps a generated behavior with a known RESTBot command.

## 4.2 The Human Interface

To reduce the interface between the user and the virtual world, we wanted to make the human operators' experience as natural as possible.

To attain a natural interface, we used a 3D TOF camera (a technology similar to Microsoft Kinect), coupled with software for tracking a users body position, resulting in the user's body becoming the controller [15].

The operational environment contains a defined physical area for the user to interact with the virtual world. When the system recognizes that a person has stepped into the scene, the user is instructed to stand still briefly while the user's body position and posture are recorded. The system then uses this information as a baseline to recognize future movements.

One of the control mechanisms implemented is the usage of moving in various directions to control avatar movement. When the user steps away from the neutral center position, it is recognized as a move in that direction, causing the avatar to move in same direction. Moving back to the original center position halts the avatar's movement. The avatar's directional movements mirror the user's movements. Step left to move the avatar left, back to move the avatar back, diagonally to move diagonally, etc.

Gestures, operator body postures and positions were combined with inputs from a Nintendo WiiMote to add some additional controls, all funneled into a modified full Second Life viewer application. The gesture system interacted with the viewer through another REST interface [14]

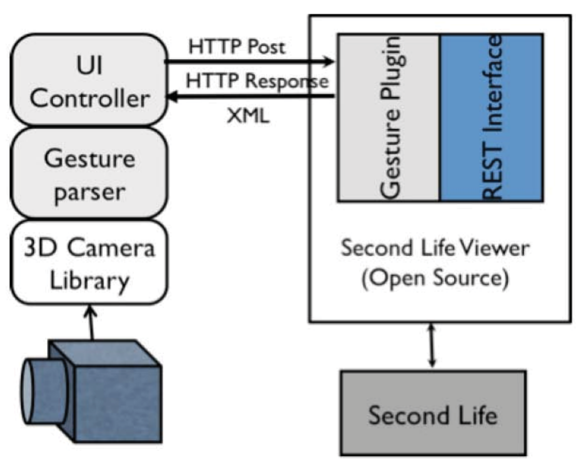


Figure 5: Architecture of User Interface

augmented with additional movement controls required to interact with our UI, controlling the avatar's actions much more directly and therefore less prone to error than having the controls merely sending keystrokes.

The viewer then displayed on a life size screen, putting the operator "into" the virtual environment and breaking another barrier to immersive experience.

## 4.3 Second Life Extensions

In order to implement a reasonably complex and immersive experience, we sourced most of the scene components from publically available vendors advertising in the Second Life Marketplace. The purchased products were assembled into a park space in one corner of a plot of virtual land we already owned.

We implemented a custom Frisbee-like game of catch for the purposes of this demonstration. While there are similar games available for purchase in Second Life already, none of the commercially available ones work well with non-human avatars, and we wanted our dog to be able to catch and carry the disc in its mouth rather than a paw.

Finally, there were a few additional minor pieces of supporting programming and building required, mainly to provide for a more seamless demonstration environment, improving overall performance, and adding a variety of sensors to improve gathering of metrics on the scenario.

To exhibit the behaviors expected from a dog, we created a few new commands for RESTBot. For example, we wanted the ability to have the dog *follow* a human controlled avatar. By adding in a new command to the RESTBot plugin library, we were able to quickly extend the tool to meet our requirements.

Once we had the ability to pass our generated behaviors and actions into the world, we hooked up our new forms of input to the behavior generator. Using BBN developed Speech recognition software we were able to translate natural language into text that was then used by the rules engine to drive the behavior. For example, we created a rule that matched the work "speak" with an action that results in the dog barking.

We also developed a gesture library for recognizing a core set of gestures, such as an arm moving in a Frisbee throwing motion. Using this physical action as input to the rules engine, we were able to generate and *throw* a Frisbee in the virtual world that would then get fetched and returned by the dog.

## 5.0 FUTURE WORK

We would like to extend the temporal reach of the AI metaphor. In particular, the strength of non-player characters is that they are always "on" and always available to interact with. We intend to approach this goal in the following significant: use Second Life's Voice-over-IP system to communicate with the bots, and enhance the AI rule engine to have an attentional model to allow it to focus on a single human avatar at a time.

First, as a matter of available time and complexity, we implemented the voice recognition system to expect a direct input from a high-quality headset. Second Life does have a functional voice communication system, but achieving high quality voice recognition of low-quality audio is a notoriously difficult problem. Furthermore, the interface to the SL voice system itself is non-trivial.

Also, we plan to enhance the AI engine and rule set to be able to focus on a single human at a time when performing. A variety of distraction behaviors as well as complex modes of interaction with multiple humans and even other AIs becomes possible.

Together, these improvements would allow us to field highly interactive AI agents around the clock to interact purely through the commodity virtual world of Second Life.

## 6.0 CONCLUSION

By decreasing the interface between a user and a virtual world and improving the behaviors of AI within the world, the overall user experience can become more immersive, allowing the user to forget about the boundary between the physical and virtual world.

In our research, we have investigated an approach to decrease the interface through the removal of the keyboard and mouse barrier. We replaced these input modalities with natural speech and gesture providing a more natural interface to the world.

We also used off-the-shelf technologies to apply more realistic behaviors to an AI avatar within a virtual world. When used in combination with the more natural interface, a user can interact with the virtual agent as if it was another entity in the physical world.

## 7.0 ACKNOWLEDGEMENTS

The authors would like to thank David Diller, Rich Shapiro and Kerry Moffitt for all of their work on the design and development of the system.

## 8.0 BIBLIOGRAPHY

1. *NetHack 3.4.3*. [cited 2011 July]; Available from: http://www.nethack.org.
2. Weizenbaum, J., *Computer power and human reason : from judgement to calculation* 1977, San Francisco: W.H. Freeman.
3. *Robots (computer game) - Wikipedia, the free encyclopedia*. [cited 2011 July]; Available from: http://en.wikipedia.org/wiki/Robots_(computer_game).
4. Güzeldere, G.a.F., S. *Dialogues with colorful Personalities of early AI*. 1995; Available from: http://www.stanford.edu/group/SHR/4-2/text/dialogues.html.
5. Carpenter, R. and J. Freeman, *Computing machinery and the individual: the personal Turing test*, 2009, Citeseer.
6. Abowd, G.D., E.D. Mynatt, and T. Rodden, *The Human Experience.* Pervasive Computing, 2002: p. 48-57.
7. *Time of Flight Camera*. August, 6, 2011 [cited 2011 July]; Available from: http://en.wikipedia.org/wiki/Time-of-flight_camera.
8. Norman, D., *The psychology of everyday things*1988, New York: Basic Books.
9. Forgy, C.L., *Rete: A fast algorithm for the many pattern/many object pattern match problem* 1.* Artificial intelligence, 1982. **19**(1): p. 17,Äì37.
10. *Drools*. [cited 2011 July]; Available from: http://www.jboss.org/drools.
11. *restbot - RESTbot allows bots in Second Life to be commanded through an HTTP interface - Google Project Hosting*. [cited 2011 July]; Available from: http://code.google.com/p/restbot.
12. Pleiades. *RESTbot Unveiled* Available from: http://pleiades.ca/2007/09/28/restbot-unveiled.
13. *libopenmetaverse developer wiki*. May 5, 2010 [cited 2011 July]; Available from: http://lib.openmetaverse.org/wiki/Main_Page.
14. Sol, D. *User:Dzonatas Sol/SNOW-375 - Second Life Wiki*. [cited 2011 July]; Available from: http://wiki.secondlife.com/wiki/User:Dzonatas_Sol/SNOW-375.
15. Diller, D., et al. *Interacting Naturally in Virtual Environments. I/ITSEC, 2010.*